Tutorial: MiniEdit 4

MiniEdit shows you how to use the more advanced features of the THINK Project Manager. You'll build a small text editor based on the sample application described in Chapter 1 of *Inside Macintosh I.* One of the source files has a small, intentional bug to show you how the THINK Project Manager makes it easy to fix mistakes.

You'll learn how to create a project, how to fix mistakes, how to run a project, how to build an application, and how to use a resource file.

Before you begin

If you didn't follow the "hello world!" tutorial in the last chapter, read it now to get an idea of how the THINK Project Manager works in general.

Make sure the `MiniEdit Folder` is on your hard disk, because it contains all the files you need to follow this tutorial. If you followed the directions in Chapter 2, "Installation," it should be in the `Demos` folder contained in your `Development` folder.

What you should know

You should know how to use the standard file dialog boxes to move around to different folders. If you don't know how to do this, read the documentation that came with your Macintosh.

Contents

Creating the Project 53
Adding the Source Files 56
Compiling and Running the Project 59
Fixing a Bug 59
Running the Project Again 62
Building the Application 64
Using a Resource File 66
Finishing Up 66
Where to Go Next 66

Creating the Project

If you have Symantec C++, you see a different folder name.

Open the folder containing the THINK Project Manager, and double-click its icon to begin. You see a dialog box that asks you to open a project, as in Figure 4-1.

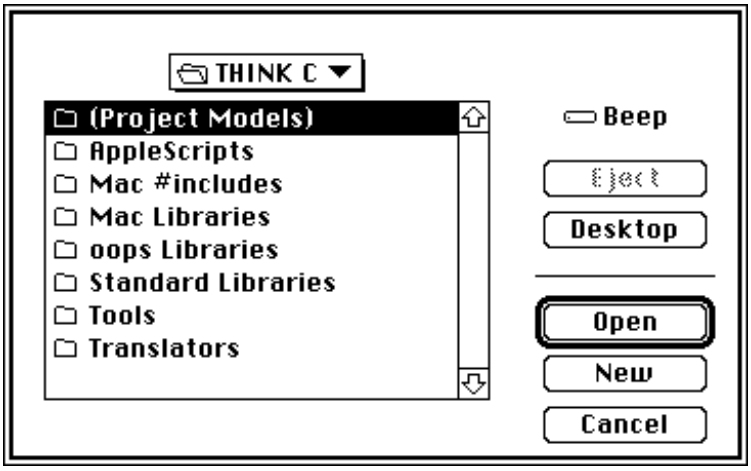

Figure 4-1 Opening a project

If you have Symantec C++, you see additional options in this dialog.

Since you're creating a new project, click the New button. You see the dialog in Figure 4-2. Since all the files you need to build this application are already in a folder included in your package, turn the

Create folder option off. Then select Mac Application Project from the list and click Create.

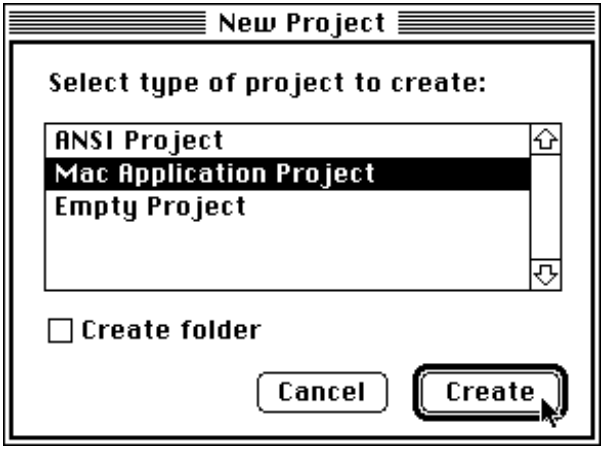

Figure 4-2 Choosing a project model

When you get the next dialog box, move to the `MiniEdit Folder` in the `Demos` folder, name your project `MiniEdit`, and click Save.

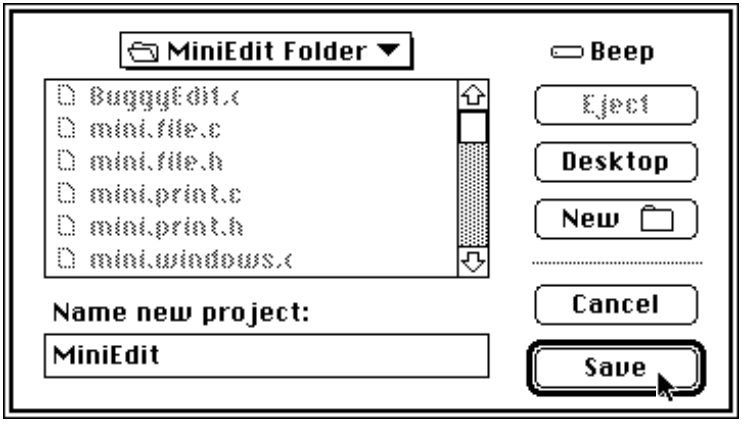

Figure 4-3 Creating a new project

The THINK Project Manager creates a new project in the `MiniEdit Folder` named `MiniEdit.π` and a project document on disk, and it displays a project window, as in Figure 4-4.

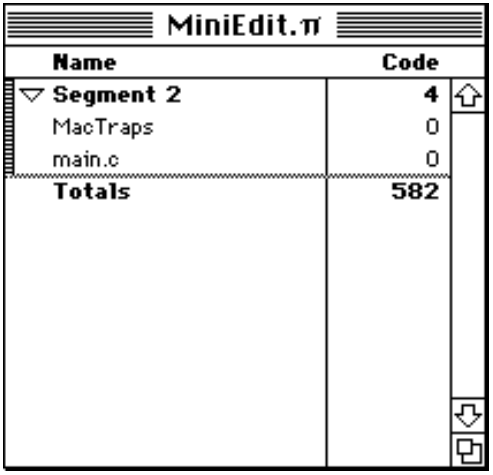

Figure 4-4 The project window for a new project file

Since all the source files you need for this application are already written, you need to remove `main.c` from the project. Just click `main.c` in the Project window, and choose **Remove** from the **Source** menu. You should also delete `main.c` by going back to the

finder and dragging `main.c` from the `MiniEdit Folder` to the Trash. Your Project window should look like that in Figure 4-5.

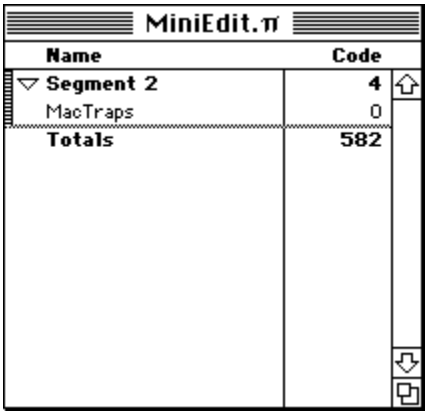

Figure 4-5 Moving the MiniEdit files

Adding the Source Files

Now you're ready to add the source files to your project. All the source files for `MiniEdit.π` are in the `MiniEdit Folder`.

Select **Add Files** from the **Source** menu. You'll see the dialog in Figure 4-6.

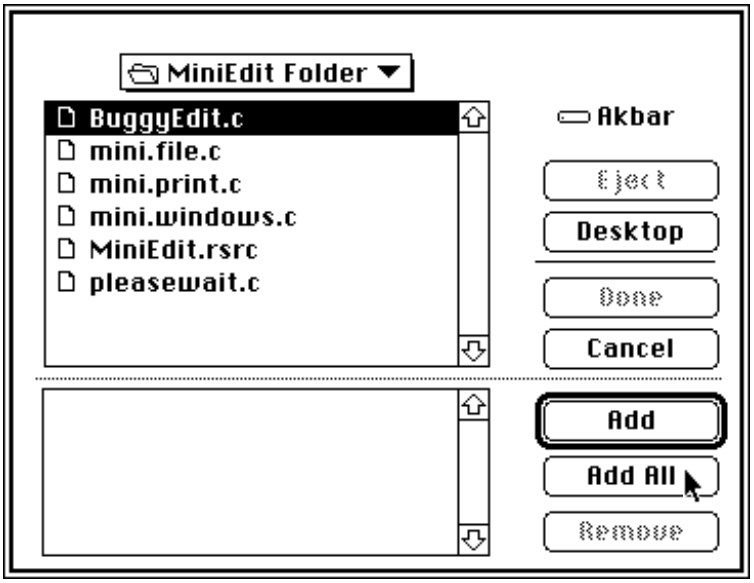

Figure 4-6 Add dialog

The top list displays the source files and libraries in the folder you're currently in. The bottom list shows the files that the THINK Project Manager will add to your project when you click Done. The Add and Add All buttons move files into the bottom list. The Remove button removes files from the bottom list.

Click the Add All button. THINK C moves all the source files in the `MiniEdit Folder` into the bottom list. The dialog should look like that in Figure 4-7.

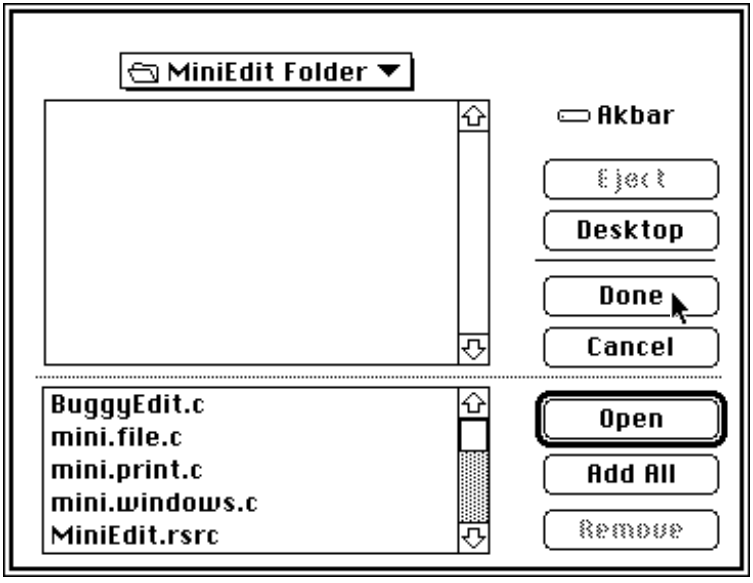

Figure 4-7 Adding the MiniEdit source files

Click the Done button now. Your project window should look like that in Figure 4-8.

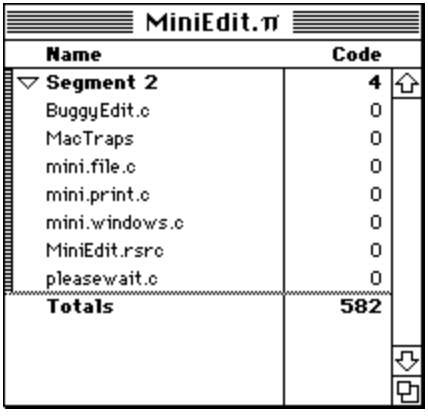

Figure 4-8 MiniEdit.π project window with all its files

The Code column displays the object size in bytes for each file. The sizes are all zero because you haven't compiled any files or loaded the MacTraps library.

Compiling and Running the Project

Before you can run your project, you need to compile the source files and load the MacTraps library. You can use the **Compile** and **Load Library** commands from the **Source** menu, or you can let the THINK Project Manager take care of everything for you.

The THINK Project Manager uses the project document to keep track of which files need to be compiled, so you can go ahead and run your project. Choose the **Run** command from the **Project** menu.

None of the files in the project has been compiled, so the THINK Project Manager asks you if you want to bring the project up to date, as shown in Figure 4-9.

Figure 4-9 Asking if you want to update your project

Click the Update button.

The THINK Project Manager starts compiling the first file in the project. It displays a dialog box that shows how many lines have been compiled. (The THINK Project Manager adds the number of lines in `#include` files in the line count.)

In this example, the THINK Project Manager opens a Compile Errors window because `BuggyEdit.c` has a small intentional bug.

Fixing a Bug

When the THINK Project Manager finds an error in your source file, it opens a Compile Errors window and displays an error message. Select the error message and press Enter to open the source file and highlight the line containing the error.

Note that the compiler continues compiling the rest of the files in the project even if it finds an error.

In Figure 4-10, the THINK Project Manager displays the message that a variable hasn't been defined.

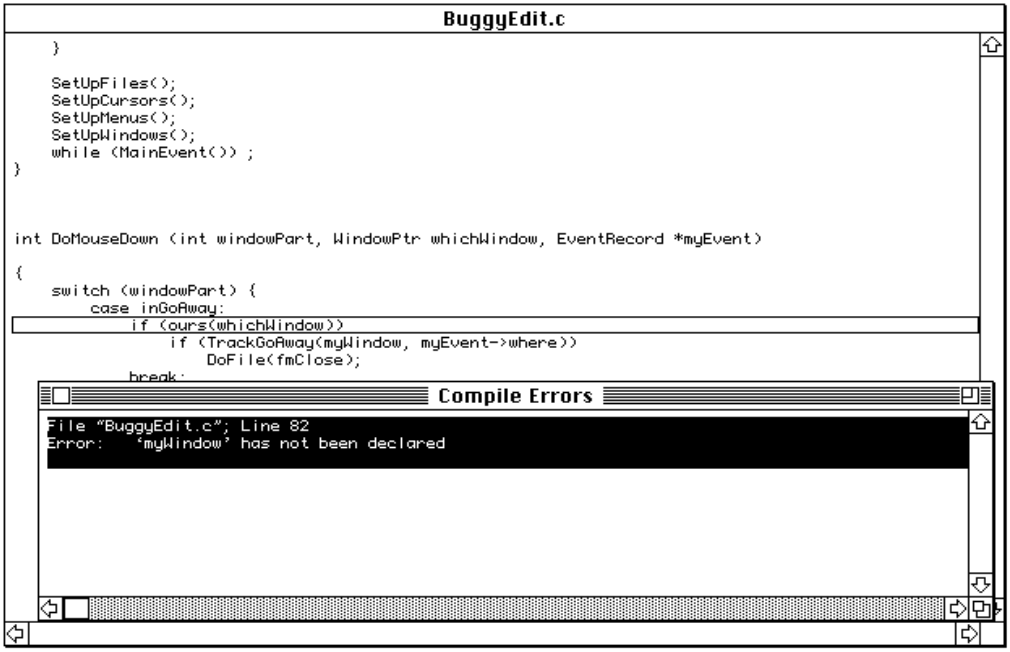

Figure 4-10 The THINK Project Manager displays an error message

Scroll toward the beginning of the file. The declaration for `myWindow` is commented out, as shown in Figure 4-11.

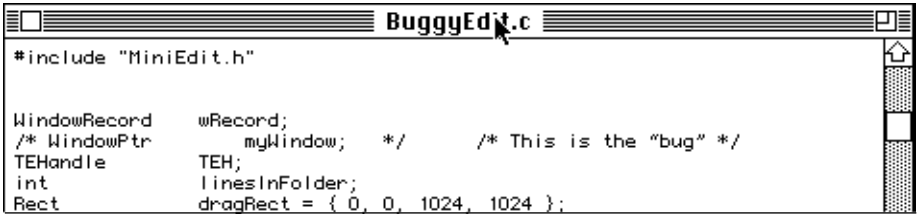

Figure 4-11 A declaration that was commented out

Remove the comments surrounding the declaration of `myWindow`.

Now compile the file `BuggyEdit.c`. Choose the **Compile** command from the **Source** menu. The THINK Project Manager will compile the source file without errors this time. Note that you don't have to save a file to recompile it.

Before you run the project again, save the changes you've made to `BuggyEdit.c`. Since the file no longer contains a bug, save it with a different name. Choose the **Save As** command from the **File** menu and save the corrected file as `MiniEdit.c`, as in Figure 4-12. (Make sure you're in the `MiniEdit Folder`.)

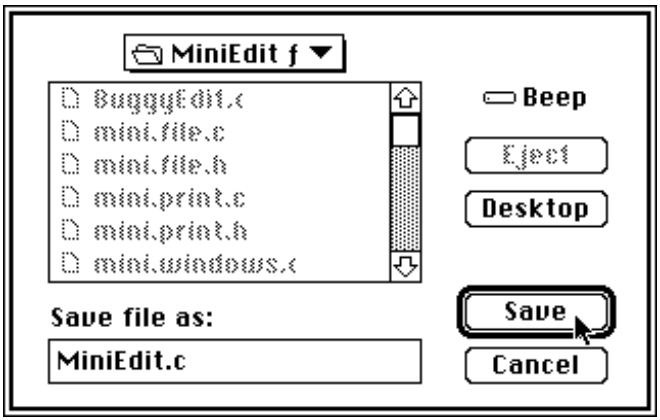

Figure 4-12 Saving a file with a new name

Now click the project window. When you use the **Save As** command on a file that is already in the project, the THINK Project Manager changes the file's name in the project window as well. The

To save a file with a different name without affecting the project, use the ***Save A Copy As*** *command.*

file's object code is now associated with the new name, as in Figure 4-13.

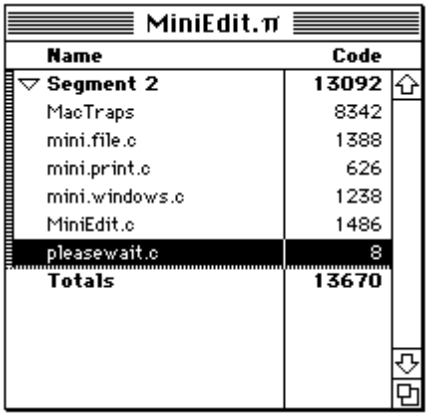

Figure 4-13 MiniEdit.π project window with a new file name

Now that you've fixed the bug, you can try running the project again.

Running the Project Again

Choose **Run** from the **Project** menu. When the THINK Project Manager asks you if you want to bring the project up to date, click the Update button.

Since the THINK Project Manager has compiled the whole project, it launches it as if you had opened it from the Finder. This is illustrated in Figure 4-14.

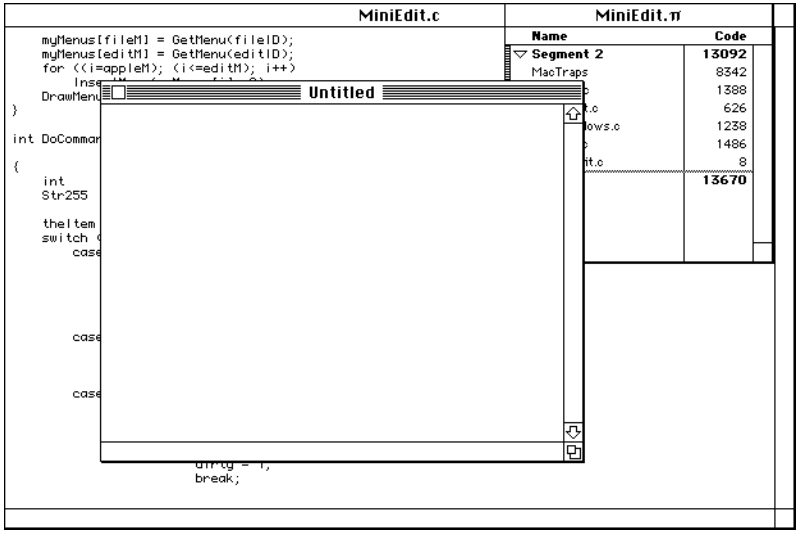

Figure 4-14 Running MiniEdit with the THINK environment

If you're using System 7 or MultiFinder under System 6, the THINK Project Manager launches your project in its own partition, so you can shift from your project back to the THINK Project Manager. If you're not using System 7 or MultiFinder under System 6, the THINK Project Manager launches your project as if you had started it from the Finder. When you quit running your project, the THINK Project Manager starts up again automatically.

Play with the MiniEdit application for a while if you like. You might want to make some changes. When you're satisfied with how the project runs, you're ready to turn it into a double-clickable application.

Building the Application

Turning your project into an application is easy. Choose the **Set Project Type** command from the **Project** menu. The dialog box is shown in Figure 4-15:

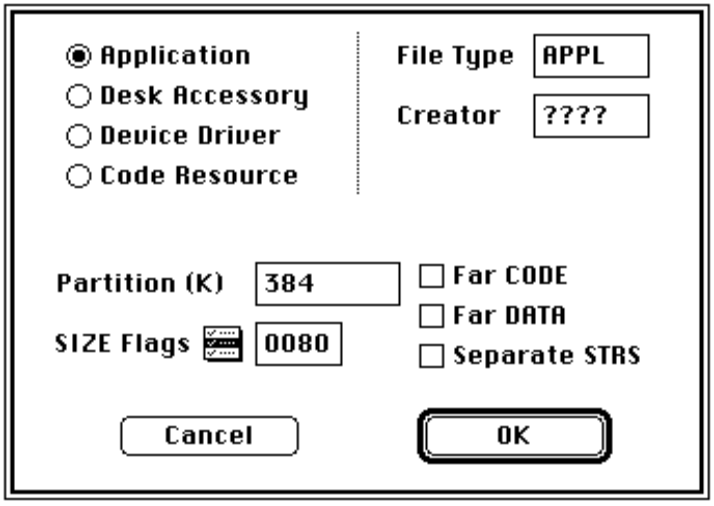

Figure 4-15 Set Project Type dialog

CEM8 *doesn't stand for anything. It was chosen because it is unlikely that any other application on your disk has that signature.*

Set the creator to CEM8. This ensures that your application has the right icon when you build it.

Set the partition size to 128K. Since MiniEdit is such a small program, it doesn't need the default 384K partition size. The Macintosh uses the partition size to determine how much memory to give to your application.

When the dialog box looks like that in Figure 4-16, click OK:

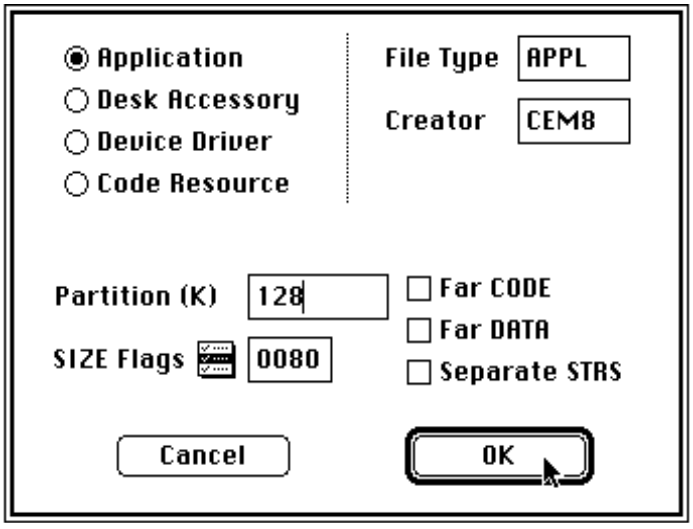

Figure 4-16 Setting the project type for MiniEdit

Choose **Build Application** from the **Project** menu. You'll see the dialog box shown in Figure 4-17.

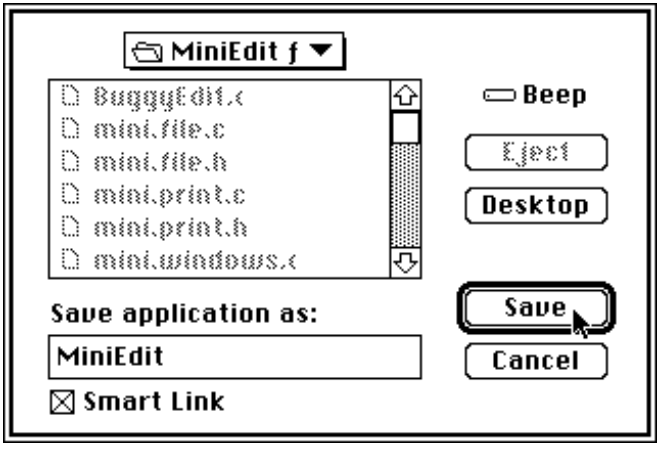

Figure 4-17 Building the MiniEdit application

Name the application `MiniEdit`. Leave the Smart Link box checked. This option tells the THINK Project Manager to make the application as small as possible.

As it's building the application, the THINK Project Manager gives you status messages. First it links all the object code. Then it copies the resource file for the project into the application. (The next section tells you how to use a resource file with a project.) When it's finished, you'll have a new application in the `MiniEdit Folder`.

Figure 4-18 The MiniEdit icon

Using a Resource File

The `MiniEdit Folder` contains a file called `MiniEdit.π.rsrc`. This file contains the resources that the MiniEdit project uses.

When the THINK Project Manager runs your project, it looks for a file named *projectname*`.rsrc`. (That is, the name of your project plus the characters `.rsrc` appended to it.) This file should contain the resources (menus, alerts, dialogs, and so on) that your project uses.

For more information on resource files, see Chapter 27, "Using THINK Rez."

To create a resource file, you can use THINK Rez or ResEdit utilities (ResEdit is included in your package). `MiniEdit.rsrc` was created with ResEdit, so there's no resource description file for it.

Finishing Up

When you're finished working on a project, you can either close the project or quit the THINK Project Manager. To close the project, use the **Close Project** command from the **Project** menu. When you close a project, the THINK Project Manager displays a dialog box that lets you open or create projects.

To quit the THINK Project Manager, choose **Quit** from the **File** menu.

Where to Go Next

The tutorial in the next chapter shows you how to use the THINK Project Manager's source-level debugger. It shows you how to activate the debugger, how to trace through your code, and how to examine and change the values of your variables.

If you feel comfortable with what you know so far, you might want to start creating your own applications right away. See the chapter "Using the THINK Project Manager" when you need help on a particular topic. You'll probably find Chapter 10, "Editing Files," useful now.

4 Tutorial: MiniEdit